h e g

Haute école de gestion
Genève

APPLICATION REPETITOR

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

par :

Diana PEDRO PINTO

Conseiller au travail de Bachelor :

Philippe DUGERDIL, professeur HES

Genève, 15 octobre 2017

Haute École de Gestion de Genève (HEG-GE)

Filière Informatique de Gestion

Hes·so GENÈVE
Haute Ecole Spécialisée
de Suisse occidentale

Déclaration

Ce travail de Bachelor est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre de Bachelor of Science en informatique de gestion.

L'étudiant atteste que son travail a été vérifié par un logiciel de détection de plagiat.

L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J'atteste avoir réalisé seule le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 15 octobre 2017

Diana Pedro Pinto

Remerciements

Un grand merci à toutes les personnes qui m'ont aidées et soutenues lors de la réalisation de ce projet, tout particulièrement :

Mikael Marques Ferreira, fondateur et mandant du projet Repetitor, pour sa confiance, son temps et son implication.

Philippe Dugerdil, professeur à la HEG et conseiller au travail de bachelor, pour sa présence, son soutien et ses bons conseils qui m'ont permis d'amener le projet à son avancée actuelle.

Solana Eduardo, chargé de cours au CUI et vacataire à la HEG, pour son temps et ses conseils en matière de sécurité.

Katty Taveira Macedo, co-fondatrice du projet Repetitor et amie, pour sa présence, son soutient et sa précieuse aide lors de la relecture et correction de ce travail.

Rolf Hauri, chargé d'enseignement, pour son aide et son temps lors de la résolution d'un problème lié à la couche de sécurité et PhoneGap

Résumé

Ce mandat a pour Object de résoudre le problème suivant : mettre en contact les étudiants de la Haute École de Gestion de Genève qui souhaitent donner des cours de soutiens et ceux qui souhaitent en recevoir.

Suite à plusieurs réunions avec le mandant, il a été décidé, de partir sur la réalisation d'une application mobile multiplateforme et multilingue mais également de la rendre disponible sur les différents Apps Stores.

Le mandant, Mikael Marques Ferreira, étudiant en Master, a remarqué que les potentiels répétiteurs et les étudiants qui souhaitent recevoir des cours ne parviennent pas à se trouver : il manque un intermédiaire.

Dans ce travail, vous pouvez voir les différentes étapes que j'ai suivie, les décisions qui ont été prises, les alternatives envisagées mais aussi une démonstration du développement actuel sous forme de Scénarios.

Tout d'abord, il a fallu identifier les différents Besoins des étudiants et déterminer leurs importances afin de trouver la meilleure solution pour eux.

Par la suite, afin de mener à bien ce mandat, de nombreuses recherches ont été effectuées à partir de ces besoins afin de mettre en avant les Solutions potentielles. Le choix final, s'est porté sur une Application Hybride qui utilisera le Framework PhoneGap. Cela permet d'obtenir une seule application multiplateforme développée avec des langages web ainsi qu'un accès aux différentes API natives de l'appareil, mais également, de faciliter le déploiement futur sur les Apps Stores.

L'une des contraintes du projet fut d'y intégrer la couche sécurité : HTTPS, en utilisant les protocoles SSL et TLS. Je décris précisément la marche à suivre que j'ai effectuée pour réaliser cet objectif dans le chapitre Sécurité.

Au moment du rendu de ce travail, j'ai développé les fonctionnalités Inscription et Configuration du profil que je détaille dans le chapitre fonctionnalités développées.

Table des matières

Déclaration i
Remerciements ii
Résumé iii
1. Introduction 1
2. Méthode de travail 2
3. Problème à résoudre 3
3.1 Qu'elle est le problème ? 3
3.2 Besoin 3
3.3 Application mobile 4
4. Solutions envisagées 5
4.1 Solution existante 5
4.1.1 Apprentus 6
4.1.1.1 Fonctionnalités proposées : 6
4.1.1.2 Point faible 6
4.1.2 Tutor24 7
4.1.2.1 Fonctionnalités proposées : 7
4.1.2.2 Point faible 7
4.2 Développement complet 8
4.2.1 Application native 8
4.2.2 Application web 9
4.2.3 Application Hybrid 9
4.2.4 CMS 10
4.3 Solution choisie 10
5. Choix de développements 11
5.1 JQuery & JQuery mobile 11
5.2 PhoneGap 12
5.3 Serveur 12
5.3.1 NodeJS 12
5.3.2 WAMP 12
5.3.3 Jersey – Web Services & REST 13
5.4 LogBack 13
6. Sécurité 14
6.1 Données 14
6.2 Encryptions 14
6.3 SSL/TLS – HTTPS 14
6.3.1 Certificat 14
6.3.1.1 JSSE 15
6.3.1.2 APR 16

6.3.2 Alternative : JCryption 16
7. Détail de la solution 17
7.1 Fonctionnalités développées 17
7.1.1 Création d'un compte 17
7.1.2 Profil 17
7.1.2.1 Onglet : Profil générale 17
7.1.2.2 Onglet : Informations sur les cours 17
7.1.3 Interface visuel 18
7.1.4 Scénarios 22
7.1.4.1 Scénario - Inscription 22
7.1.4.2 Scénario 2 : Configuration du profil 23
7.1.5 Architecture du code 26
7.2 Fonctionnalités à développer 27
7.2.1 Disponibilité 27
7.2.2 Recherche 27
7.2.3 Prendre / proposer un rendez-vous 27
7.2.4 Consulter l'agenda 27
7.2.5 Favoris 27
7.2.6 Tchat 27
7.2.7 Réputation 27
7.2.8 Payement 27
7.2.9 Contact et proposition 28
8. Conclusion 28
Bibliography 29
Annexes 1
Annexe 1 – Ebauche du diagramme de classe 1
Annexe 2 – Diagramme de classe actuel 2
Annexe 3 – Use Case 3
Annexe 4 – Les Works flow identifiés 4

Liste des figures

Figure 1 : Global Internet Users 4
Figure 2 : Internet Usage (Engagement) = Solid Growth 4
Figure 3 : … eCommerce Growth 5

1. Introduction

La Haute École de gestion de Genève (HEG) est une école de niveau universitaire qui compte quatre formations Bachelor et trois en Master. Elle fait partie des 28 écoles de la Haute école spécialisée de Suisse occidentale (HES-SO).

À la rentrée 2017-2018, La Haute École de Gestion compte un total de plus de 1440 étudiants dont le nombre augmente chaque année.

Une école qui évolue

En effet, la HEG fait tout pour s'améliorer et évoluer d'années en années que ce soit au niveau de la mise à jour du contenu de ses cours, l'amélioration des systèmes informatiques, la mise en place d'espaces de travail pour les étudiants et même un nouveau bâtiment en 2016.

Il y a quelques mois, le mandant a constaté que les étudiants qui souhaitent prendre des cours de soutient avaient beaucoup de mal à trouver un répétiteur pour les aider. Pareil pour les potentiels répétiteurs qui souhaitent donner des cours afin de gagner un peu d'argent. Actuellement, que font-ils ?

Au semestre précédent, nous avons constaté que plusieurs d'entre eux posaient des annonces sur les tableaux d'informations disponibles dans les différents bâtiments. Résultats, les languettes de papier prédécoupées avec le numéro de l'annonceur partaient mais n'aboutissaient qu'à très peu, voire pas du tout, de réponses.

Comment répondre à ce problème ?

2. Méthode de travail

Afin de répondre au mieux à cette question, la première étape fut de regarder avec le mandant quels sont les besoins que l'application doit couvrir. De quoi les étudiants ont besoin en priorité et qu'est-ce que notre solution peut apporter de plus.

Savoir ou on met les pieds !

Dès que le problème et les besoins furent définis et validés par le mandant, de nombreuse recherches ont débuté. Tout d'abord, autour des diverses solutions possibles. Dès le début du projet, le mandant voulait partir sur une application mobile ce qui représentait déjà un avantage en soit car aucune application mobile de ce genre n'existe, actuellement pas, en Suisse. Mais il souhaitait partir sur plusieurs développements natifs. Dans un premier temps, je lui ai conseillé de partir sur une version beta avec une unique application hybride. Cela permettait un développement plus rapide et moins couteux. Ce qu'il a accepté.

Pour les choix de développement, le mandant n'avait pas de préférences particulières, j'ai donc eu carte blanche pour faire mes proposes choix tout en tenant compte des différents besoins et objectifs du projet.

Mes choix se sont portés sur des technologies que je connaissais déjà et qui m'intéressent : JQuery mobile, Service Web REST Java. Mais j'en ai également intégré de nouvelles avec PhoneGap, LogBack et SSL/TLS pour la sécurité.

Tout au long du projet, il était important de toujours garder contact avec le mandant et de le voir régulièrement pour lui montrer l'avancement et confirmer que la direction prise, est là bonne. Dès que mes premières maquettes html étaient prêtes, j'ai organisé une réunion pour valider celles-ci. Dès qu'une fonctionnalité étais prête, une autre rencontre était organisée et ainsi de suite.

Des réunions hebdomadaires étaient, également, organisées avec le directeur de mémoire chaque semaine. Elles permirent de faire un suivie du travail réalisé et d'obtenir de nombreux conseils qui m'ont souvent débloqué.

Tout cela m'a permis de développer les fonctionnalités Inscription et Configuration du profil. Ainsi que de mettre en place la couche de sécurité HTTPS.

3. Problème à résoudre

3.1 Qu'elle est le problème ?

Comme expliqué précédemment, il manque un intermédiaire entre les étudiants qui souhaitent donner des cours de soutiens et ceux qui souhaitent en recevoir. Ils doivent pouvoirs se trouver et se contacter afin organiser leurs cours en fonction des horaires de chacun.

La cible est donc tous les étudiants de la Haute École de Gestion de Genève, ce qui comprend quatre filières de Bachelor.

La solution choisi doit, absolument, être disponible pour tous les étudiants !

3.2 Besoin

Avant de essayer de résoudre le problème, il est important de trouver et de comprendre les besoins des étudiants afin d'y répondre du mieux possible :

Parmi les quatre filières de la HEG, nous avons l'International Business Management, dont les cours sont donnés en anglais. De plus, l'école accueille de nombreux étudiants étrangers venant passer un semestre dans l'école. Il est donc important que la solution soit multilingue (au moins français et anglais).

Entre les cours, les révisions et le travail, les étudiants ont généralement peu de temps libre. L'un des points les plus importants est d'avoir un système facile d'utilisation, simple avec des processus rapides.

Un étudiant doit avoir la possibilité de recevoir et de donner des cours, s'il le souhaite.

Un répétiteur doit avoir la possibilité de noter ses disponibilités afin de permettre à un élève de proposer un cours à une heure qui, normalement conviendrait aux deux.

Les utilisateurs doivent pouvoir faire une recherche adaptée pour leurs permettre de trouver le répétiteur qu'ils leurs faut.

La solution sera un point de contact entre les étudiants qui souhaitent donner des cours et ceux qui souhaitent en recevoir, il faut donc prévoir un système de communication entre eux.

Un petit plus serait de laisser le choix aux étudiants, qui le souhaitent, de payer via l'application, pour leur permettre de gagner du temps et de simplifier la gestion de leurs cours.

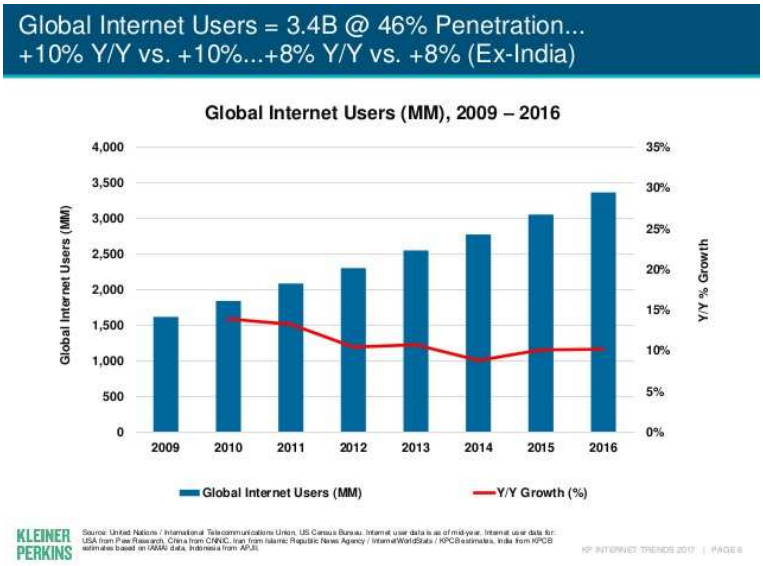
3.3 Application mobile

L'un des principaux besoins est que l'application doit absolument être disponible pour tous les étudiants de la HEG. Après discusion avec le mandant, il a été décidé de partir sur une solution mobile mais *pourquoi ce choix ? Qu'est ce que cela apporte ?*

Le 31 mai 2017, Mary Meeker présente le document **Internet trends 2017**.

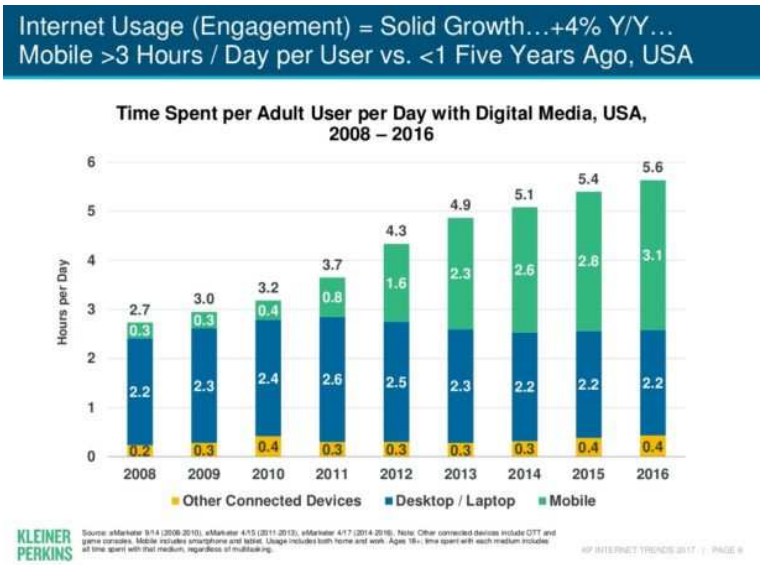
D'après celui-ci, l'engouement pour le mobile ne cesse de croître. Le nombre global d'internautes ne cesse augmenter. De même que le temps passé sur les smartphones.

Figure 1 : Global Internet Users



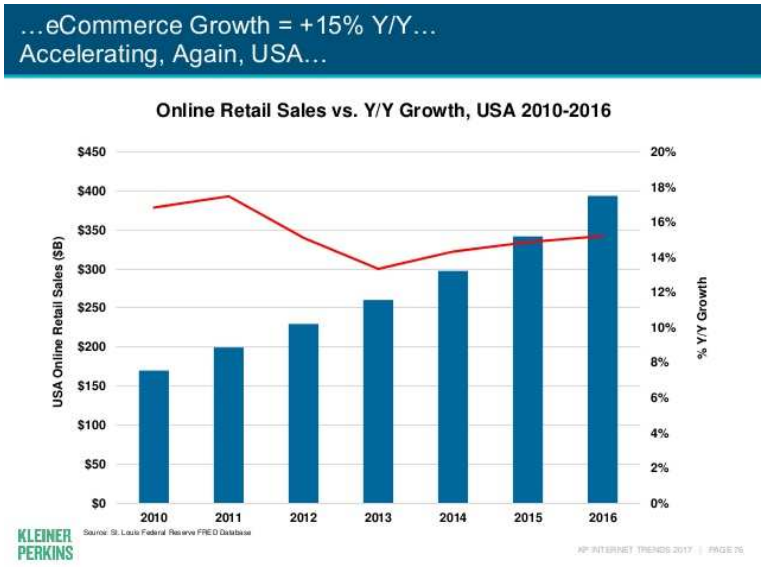
Depuis 2014, nos amis américains passent plus de temps sur mobile que sur ordinateur. Presque 1 heure de plus en 2016.

Figure 2 : Internet Usage (Engagement) = Solid Growth



Aujourd'hui, on peut tout faire avec notre smartphone, ou presque. Même faire les courses en lignes devient, petit à petit, une part de notre quotidien : +15% de croissance dans le domaine de eCommerce.

Figure 3 : … eCommerce Growth



Les applications mobiles représentent un véritable plus car elles permettent à leurs utilisateurs d'y avoir accès en tout temps avec leurs smartphones.

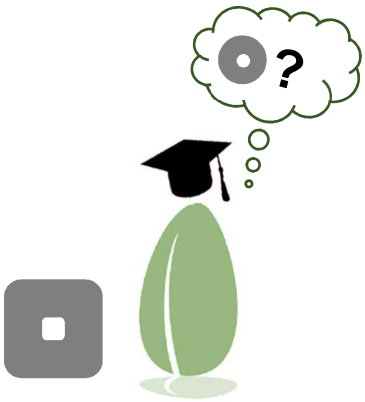
4. Solutions envisagées

Il a donc été décidé de partir sur le développement d'une application mobile pour répondre aux différents besoins des étudiants. L'application s'appelera Repetitor.

Pour cela, plusieurs solutions sont possibles :

4.1 Solution existante

Ne pas réinventer la roue !



La solution la moins coûteuse et la plus rapide à mettre en place est de trouver un système existant qui répondrait complétement ou partiellement aux divers besoins que nous avons identifiés précédemment. Adapter un système qui existe déjà permettrait de gagner du temps de développement fournirait une solution rapidement.

Deux sites internet sont remontés parmi les diverses recherches effectuées :

4.1.1 Apprentus

Site disponible pour de nombreux pays, dont la Suisse. Permet de trouver des professeurs particuliers pour des cours de musique, cours de langue, soutien scolaire à domicile et autre.

4.1.1.1 Fonctionnalités proposées :

- Créer profil
- Rechercher un cours
- Voir disponibilité d'un répétiteur
- Poser une question
- Système de favoris
- Newsletter
- Permet le choix de la devise (paiement)
- Possibilité de reporter ou annuler un cours
- Gérer les paiements par le site
- Assistance à la clientèle 7/7j
- Système de recommandation (notation)
- Inscription gratuite pour les professeurs
- La garantie bon-prof est accordée si le professeur accepte de verser une commission lors de ses cours
- Agenda des cours

4.1.1.2 Point faible

- Paiement de frais bancaires lors des transactions (en Suisse)
- Toutes les alertes passent par mail. Problème : certains emails ne passent pas, ce qui fait que le répétiteur manque des "contrats"
- Impossible de donner le numéro personnel par le site (blocage de la part du site)

4.1.2 Tutor24

Site développé et hébergé en Suisse permettant de trouver répétiteur ou élève via un système d'annonce sous forme d'offre d'emploi.

4.1.2.1 Fonctionnalités proposées :

- Créer un profil élève
- Créer un profil prof
- Rechercher un prof ou élève
- Recherche personnalisée (distance, matière…)
- Consulter les disponibilités des répétiteurs par semaine
- Noter un répétiteur

4.1.2.2 Point faible

- Un compte est soit élève, soit répétiteur. Si on souhaite être les deux, il faut créer deux comptes.
- Pour les comptes élèves, la prise de contact via le site exige un compte prémium à partir de CHF 7.50 par mois (prendre contact avec des répétiteurs, lire les messages, répondre…)

Nous avons également trouvé quelques applications mobiles disponibles pour l'Espagne et la France mais les notes et les commentaires n'étant pas excellent, ils ont été rapidement mis de côté.

4.2 Développement complet

La deuxième solution (la plus couteuse) est de partir sur un développement complet mais également complètement personnalisé et adapté.

Pour cela, plusieurs possibilités sont possibles :

4.2.1 Application native

Ces applications sont développées pour une plateforme spécifique avec un langage de programmation qui leur est propre :

- iPhone : Swift (avant cela Objective-C)
- Android : Java
- BlackBerry : Java
- Windows Phone : C# ou VB.NET

Pour pouvoir utiliser une application native, il faut la télécharger, à partir de la boutique en ligne correspondante et l'installer.

Ces applications ont de nombreux **avantages** :

- Permettre l'accès aux différentes API de l'appareil comme l'appareil photo, le GPS, les capteurs et autres.
- Permettre l'utilisation de l'application sans connexion internet.
- Application parfaitement adaptée à l'appareil engendrant de hautes performances et une App rapide.

Mais également, des **désavantages** :

- Un temps de développement plus long (une application à développer par système différents)
- Coûts supplémentaires (Plusieurs applications à maintenir en parallèle, avec des langages différents)

Quand utiliser ce type d'application ?

Pour les jeux, le meilleur choix est de réaliser une application native pour des questions de performance et de rapidité.

Dans l'idéal, si le budget, les ressources et le temps le permettent, il est préférable de partir sur du natif.

4.2.2 Application web

Contrairement aux applications natives, les WebApps sont accessibles via les navigateurs mobiles et ce, peu importe la plateforme, d'où sa force. Une application Web est donc multiplateforme.

Ces applications ont de nombreux **avantages** :

- Développement rapide (une seule application à développer)
- Multiplateforme
- Moins coûteuses (un seul développement, une seule application à maintenir)
- Application web donc basée sur les langages de programmation HTML5, CSS3 et JavaScript (moins de langage à connaître)
- Application responsive

Mais également, des **désavantages** :

- Ne peut fonctionner sans connexion internet.
- Ne peut être distribuée via stores.
- Ne permet pas l'accès aux différentes API de l'appareil

La tendance pour les applications web fait que leurs performances ne cessent d'être améliorées. HTML5 permet déjà l'utilisation de certaines API comme le GPS.

4.2.3 Application Hybrid

Le terme Hybrid qualifie un organisme issu du croisement entre deux éléments de différentes natures. Dans ce cas-ci, un mélange de l'application native et de la WebApps : un développement déployable sur tous les stores.

Pour cela, il faut intégrer un package natif comme PhoneGap qui va envelopper un navigateur dans une application native et afficher l'application web à travers elle.

Quand utiliser ce type d'application ?

Les applications Web ou Hybrid sont idéales si l'application doit fonctionner sur toutes les plateformes et tablettes. Mais également, si le développement doit être rapide et à moindre coût.

4.2.4 CMS

Les CMS, ou système de gestion de contenu, sont des logiciels destinés à la conception de site web. Ils facilitent la conception, la mise à jour et gèrent automatiquement l'affichage des pages web, leurs contenus et la liaison avec la base de données. Ils permettent également de réaliser un site web sans connaissances en programmation via des interfaces graphiques. Les principeaux problèmes des CMS est qu'ils manquent de fléxibilité, sont moins performants et ont plus de chance de se faire attaquer.

Dans un premier temps, j'ai réalisé de nombreuses recherches et tests afin de trouver lequel pourraient être le plus compatible. Plusieurs ont retenu mon attention comme Siberiane et Mobincube pour créer des applications mobile ainsi que Drupal et Wordpress pour réaliser un site web.

Le seul CMS qui permettait d'implémenter toutes fonctionnalités voulues, ou presque, gratuitement était Drupal mais sa complexité et le manque de pratique pourraient poser problème.

4.3 Solution choisie

Après discussion avec le mandant, nous avons choisi de partir sur une version Beta du projet avec un développement Hybrid complètement personnalisé. Ce qui permet d'avoir les avantages d'une application web : système multiplateforme, moins coûteux, mais également d'avoir accès aux API de l'appareil.

Au vu de ce choix, le côté client est donc complètement développé avec des langages web dont HTML, CSS, JavaScript et Ajax. Mais pas seulement comme nous allons le voir dans la suite de ce document.

Concernant la base de données, après avoir fait une ébauche du diagramme de classe (Annexe 1), j'ai choisi de partir sur une base MySQL. J'ai fait ce choix parce que j'ai déjà travaillé sur plusieurs projets qui l'utilisaient et qu'il permet un traitement simple des données.

5. Choix de développements

Tout au long de ce travail, j'ai dû faire des choix, que ce soit pour le côté client ou le côté serveur. Mes choix se sont portés sur l'intégration de PhoneGap et Jquery mobile du côté clients afin de réaliser l'objectif qui est une application hybride. Du côté serveur, mon choix s'est porté sur Jersey et la réalisation de Web Service Java avec l'intégation de Logback pour la gestion d'événement.

Dans ce chapitre, j'explique les diverses possibilités identifiées et choisis pour le développement de mon client et de mon serveur.

5.1 JQuery & JQuery mobile

En plus des langages web traditionnels (HTML, CSS, JavaScript), j'ai choisi d'intégrer JQuery et JQuery mobile.

JQuery est une des bibliothèques JavaScript les plus utilisées. Elle est openSource et a été créée pour faciliter le code côté client lors du développement d'un site internet.

"Write less, do more" → *Écrire moins pour faire plus.*

Cette devise résume bien ce que JQuery permet de faire.

Exemple :

JavaScript :

```
var listItems = document.querySelectorAll('li');
var i;
for (i = 0; i < listItems.length; i++) {
    listItems[i].className = 'starred';
}
```

=

JQuery :

```
$("li").addClass("starred");
```

Au jour d'aujourd'hui, de plus en plus de personnes naviguent sur internet en utilisant un smartphone. Pour ne pas rester à la traîne, les entreprises doivent se diriger progressivement vers des sites mobiles pour s'adapter à toutes les plateformes. C'est là qu'intervient JQuery mobile. Il s'agit d'un Framework JavaScript dont l'objectif est de permettre au développeur de réaliser rapidement des applications mobiles.

La popularité de JQuery et JQuery Mobile a entraîné la création de nombreux plugins très sympathiques qui permettent de simplifier le développement de certaines fonctionnalités

comme les plannings avec DHTMLX Scheduler, mMenu pour réaliser des menus, HighchartTable pour faire des graphiques et plus encore.

Le site officiel de JQuery et JQuery mobile offre la possibilité d'utiliser un éditeur de thème : ThemeRoller. Il permet de personnaliser l'application facilement et à notre goût.

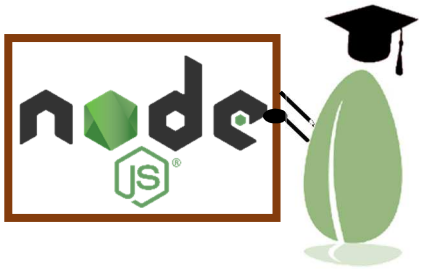
5.2 PhoneGap

PhoneGap est un Framework open source permettant de réaliser des applications hybrides à partir de langage web (HTML, CSS, JavaScript). Il fournit une API JavaScript qui permet d'accéder aux fonctions natives de l'appareil et facilite la distribution de l'application sur les différents App Stores.

Pour réaliser une application hybride, il faut utiliser un packager natif, pour Repetitor, j'ai choisi celui-ci. Il va compiler le code de l'App pour chaque plateforme qui pourra, ensuite, être interprété avec le navigateur web de l'appareil. Il donnera l'impression d'avoir une application native. Ce Framework est églament compatible avec Jquery mobile, avec qui il est souvent utilisé.

5.3 Serveur

5.3.1 NodeJS



NodeJS est une plateforme logicielle libre contenant une bibliothèque de serveur http qui permet de faire tourner un serveur web. NodeJS travaille sur un environnement bas niveau ce qui lui permet d'exécuter du code JavaScript du côté serveur ; nous avons donc l'avantage d'avoir le même langage côté client et serveur. Le manque de connaisance et de pratique de cet environnement peut poser problème.

5.3.2 WAMP

L'architecture Wamp, Windows Apache MySQL Php, est principalement utilisée pour développer des sites internet. Comme l'indique son nom, elle utilise un serveur web **Apache** pour répondre aux requêtes du client web, les données sont gérées et stockées par **MySQL**, **PHP** est le langage le plus souvent utilisé mais il est également possible de voir du Perl ou du Python.

Cette architecture est généralement utilisée sur **Windows** qui gère le tout.

5.3.3 Jersey – Web Services & REST

Jersey est un Framework open-source créé pour simplifier le développement des Web Services RESTful en Java.

Pour cela, il fournit une sorte de boîte à outils qui se base sur les API JAX-RS avec des fonctionnalités et des utilitaires qui lui sont propres.

J'ai choisi de développer le côté serveur avec Jersey et REST Java pour deux raison :

Premièrement, dans le cadre du cours 626-1, j'ai réalisé de nombreux TPs sur le sujet et aquis un semestre d'expérience. Ce qui fait que la création, la mise à en place du serveur, de la base de données ainsi que le développement des Web Service, se sont tous déroulés rapidement.

Les Web Services sont de plus en plus utilisés. Personnellement, je trouve que c'est vraiment un sujet très interessant, avec lequel j'ai pris beaucoup de plaisir à travailler.

Les seuls problèmes que je n'ai pas traités lors du cours 626-1 étaient la mise en place de la sécurité avec SSL et TLS ainsi que la mise en place d'headers dans mes réponses, afin d'eviter les erreurs de Cross-Origing Resource Sharing (CORS).

5.4 LogBack

C'est quoi le logging et à quoi ça sert ?

Logging ou historique des événements représentent la liste des différents événements, exception survenant dans une application informatique. Cela permet de comprendre le suivi des activités internes d'un processus, de déboguer, de rechercher la source des anomalies mais également de comprendre les flux de traitements. Il est toujours conseillé d'utiliser un loggeur mais plus l'application est conséquente, plus son importance est grande.

Dans cette application, j'ai choisi d'utiliser l'architecture LogBack qui succède à Log4j. Ayant le même développeur, ils sont très semblables au niveau du fonctionnement mais LogBack rassemble toutes les recherches et expériences du logging actuel. Il est très rapide et suffisamment générique pour être appliqué dans de nombreuses situations.

6. Sécurité

6.1 Données

Afin de conserver l'intégrité de la base de donneés, il est important de toujours tester toutes les données que l'utilisateur entre dans l'application ainsi que les données transmises au serveur.

6.2 Encryptions

Les mots de passe ne doivent pas être enregistrés en clair dans la base de données. Ils doivent être cryptés à l'aide de l'algorithme de notre choix. Lors de la connexion, on crypte le mot de passe rentré par l'utilisateur à l'aide du même algorithme avant de le comparer à l'enregistrement de la base de données.

Pour l'application Repetitor, j'ai choisi SHA256 (Secure Hash Algorithm) de la librairie crypto-js. Cette algorithme génère une chaîne de caractères, presque unique, fixe de 256 bit et ne peut être décryptée.

Ex : café → 850f7dc43910ff890f8879c0ed26fe697c93a067ad93a7d50f466a7028a9bf4e

6.3 SSL/TLS – HTTPS

SSL et son successeur TLS sont des protocoles de sécurité permettant des échanges de données sécurisées. Pour cela, il va crypter et protéger toutes les communications et données entre le client et le serveur. Cela évite qu'une tierce personne ait accès et modifie ces données.

Ces protocoles permettent d'éviter une attaque de type *man-in-the-middle.* Ce type d'attaque consite à intersepter les données qui transite entre un client et un serveur.

Les objectifs de sécurité pour ces protocoles sont :

- L'authentification du serveur et client (si besoin)
- La confidentialité et l'intégrité des données et des messages

6.3.1 Certificat

L'authentification se fait par certificat numérique qui permet d'identifier et certifier l'authenticité du serveur et/ou du client.

Comment l'appliquer à un serveur Tomcat ?

La première étape est de créer le fichier de la clé privée du serveur et le certificat d'authentification. Pour le développement, j'utilise un certificat auto signé mais ce n'est pas du tout recommencer pour une version publiée.

Tomcat fonctionne uniquement avec format de clé JKS, PKCS11 ou PKCS12. JKS est le format standard créé avec les lignes de commande.

Par défaut, chaque entrée est identifiée par un alias (dans mon cas tomcat).

Windows: `"%JAVA_HOME%\bin\keytool" -genkey -alias tomcat -keyalg RSA -keystore \path\to\my\keystore`

Unix: `$JAVA_HOME/bin/keytool -genkey -alias tomcat -keyalg RSA -keystore /path/to/my/keystore`

Par défaut, le mot de passe est "*changit*".

Vous pouvez le changer mais n'oubliez pas de spécifier ce changement dans le fichier *$CATALINA_BASE/conf/server.xml.*

Ensuite, il faut configurer le serveur. Il y a deux implémentations différentes de SSL sur Tomcat :

6.3.1.1 JSSE

Pour la version de développement, j'ai choisi JSSE car il est intégré à Java (depuis la version 1.4) et est donc plus rapide à mettre en place. Cette solution fonctionne très bien pour un trafic bas ou moyennement important.

Les attributs de configuration pour SSL sont différents selon l'implémentation, il est donc recommandé d'éviter l'implémentation automatique. Pour cela, il est important de spécifier le nom de la classe dans le connecteur. En JSSE, il faut adapter les connecteurs dans le fichier *$CATALINA_BASE/conf/server.xml* avec :

```
<Connector connectionTimeout="20000" port="8080" protocol="HTTP/1.1" redirectPort="8443"/>

<Connector SSLEnabled="true" clientAuth="false" keystoreFile="[emplacement de votre
clé]\key\mykey.keystore" keystorePass="changeit" maxThreads="150" port="8443"
protocol="org.apache.coyote.http11.Http11NioProtocol" scheme="https" secure="true"
sslProtocol="TLS"/>
```

Après avoir redémarré le serveur, la mise en place du protocole SSL avec l'implémentation JSSE est terminé et il est possible de le tester avec :

http://localhost:8443

6.3.1.2 APR

APR utilise OpenSSL dans la majorité des cas. Cette solution demande plus de configuration et de test. Mais dans l'idéal, il faudra mettre en place cette configuration sur le serveur officiel lors de la publication de l'application car il permet plus de trafic, peut être optimisé pour être ainsi plus rapide et consommer moins de ressources.

6.3.2 Alternative : JCryption

J'ai préféré implémenter SSL/TLS dans mon serveur Tomcat tout simplement parce que il s'agit de la base en matière de sécurité et que je n'avais jamais mis cela en place auparavant. Cependant, j'ai trouvé une alternative intéressante : un plugin JQuery nommé JCryption.

JCryption est un plugin qui utilise l'algorithme RSA et permet de crypter les communications client-serveur. Il est compatible avec OpenSSL et utilise deux librairies JavaScript : CryptoJS et JSEcrypt.

Il est très simple à mettre en place du côté client :

```
$(function() {
    $("form").jCryption();
});
```

Du côté serveur, PHP4 fournit une classe permettant de gérer la récupération des données.

7. Détail de la solution

7.1 Fonctionnalités développées

7.1.1 Création d'un compte

Afin d'accéder à l'application et d'utiliser les différentes fonctionnalités proposées par celles-ci, les utilisateurs doivent absolument créer un compte simple. Pour cela, les informations à remplir sont :

- **Nom, Prénom, E-mail :** Permet d'identifier l'utilisateur.
- **Mot de passe :** Permet à l'utilisateur de s'identifier sur l'application.

7.1.2 Profil

Il est important de donner l'opportunité aux répétiteurs de ce démarquer des autres, pour cela, ils ont la possibilité de remplir leurs profils. Cela permettra de les trouver et de pouvoir organiser des rendez-vous de cours.

Cette partie est divisée en trois sous-parties : Profil général, Informations sur les cours et disponibilités (cette dernière n'a pas été développée à ce stade).

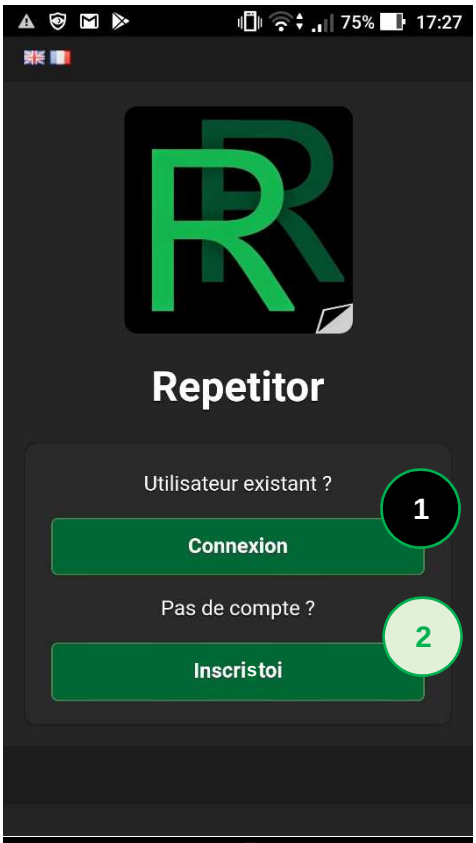
7.1.2.1 Onglet : Profil générale

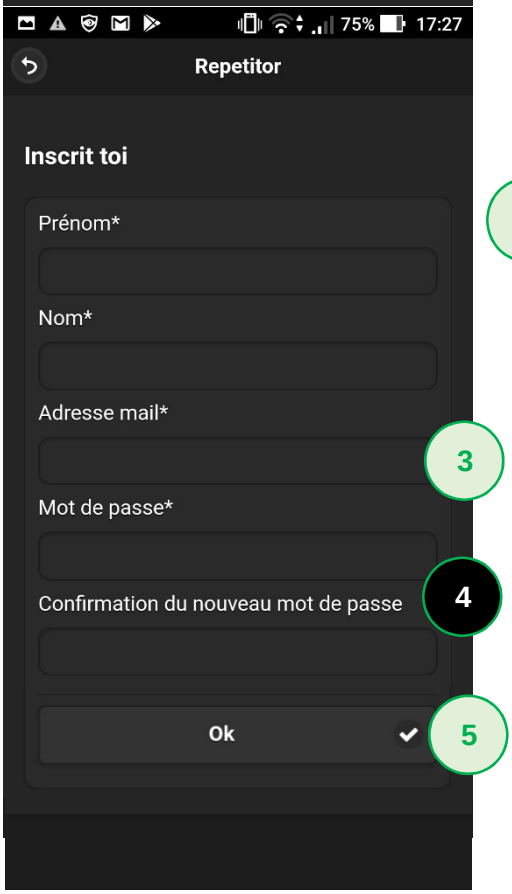
- **Nom, Prénom, E-mail :** Identifier le répétiteur
- **Filière :** Filtrer la recherche
- **Lien LinkedIn, blog ou portfolio**
- **Description :** *"Vendez-vous !"*

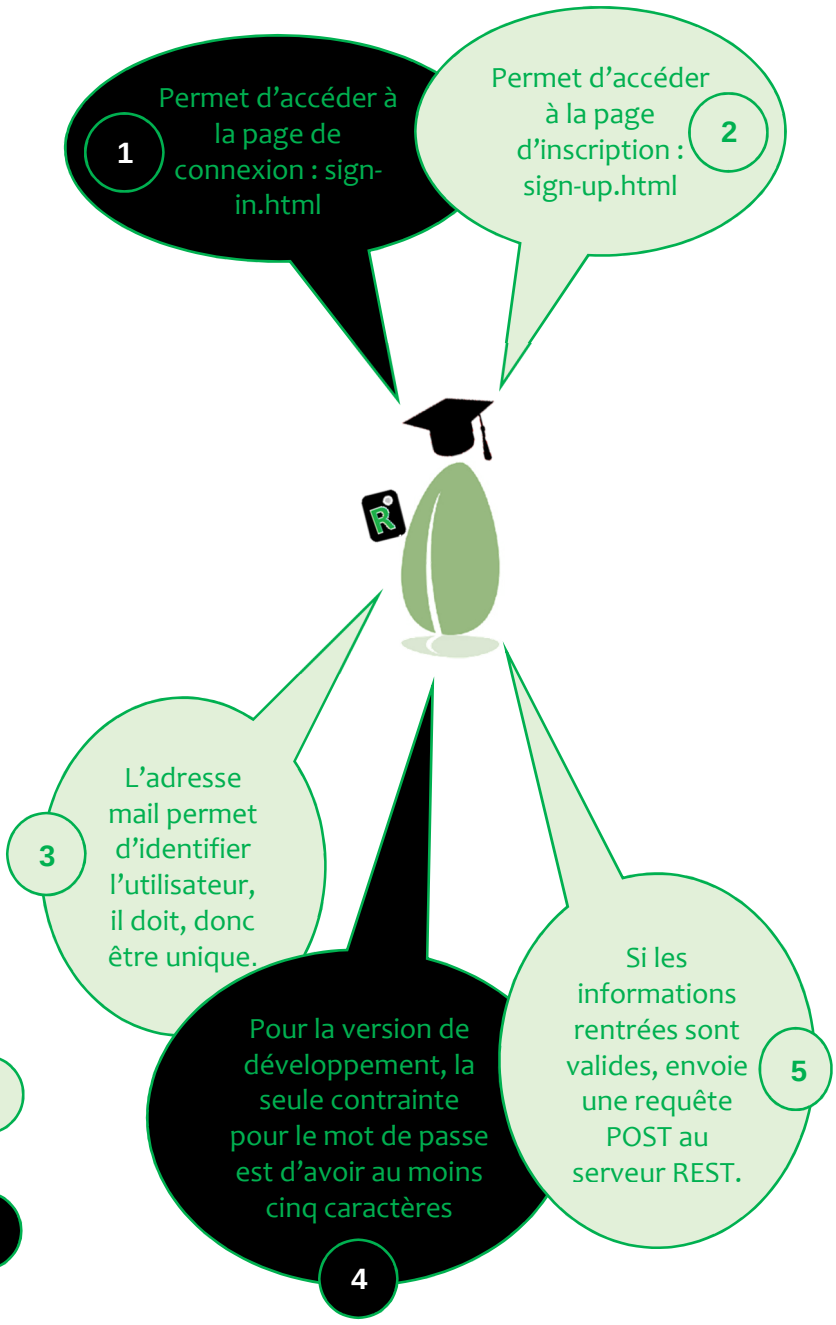
7.1.2.2 Onglet : Informations sur les cours

- **Prix et lieu du cours :** Modalité du cours.
- **Cours :** Liste des cours donnés par le répétiteur.

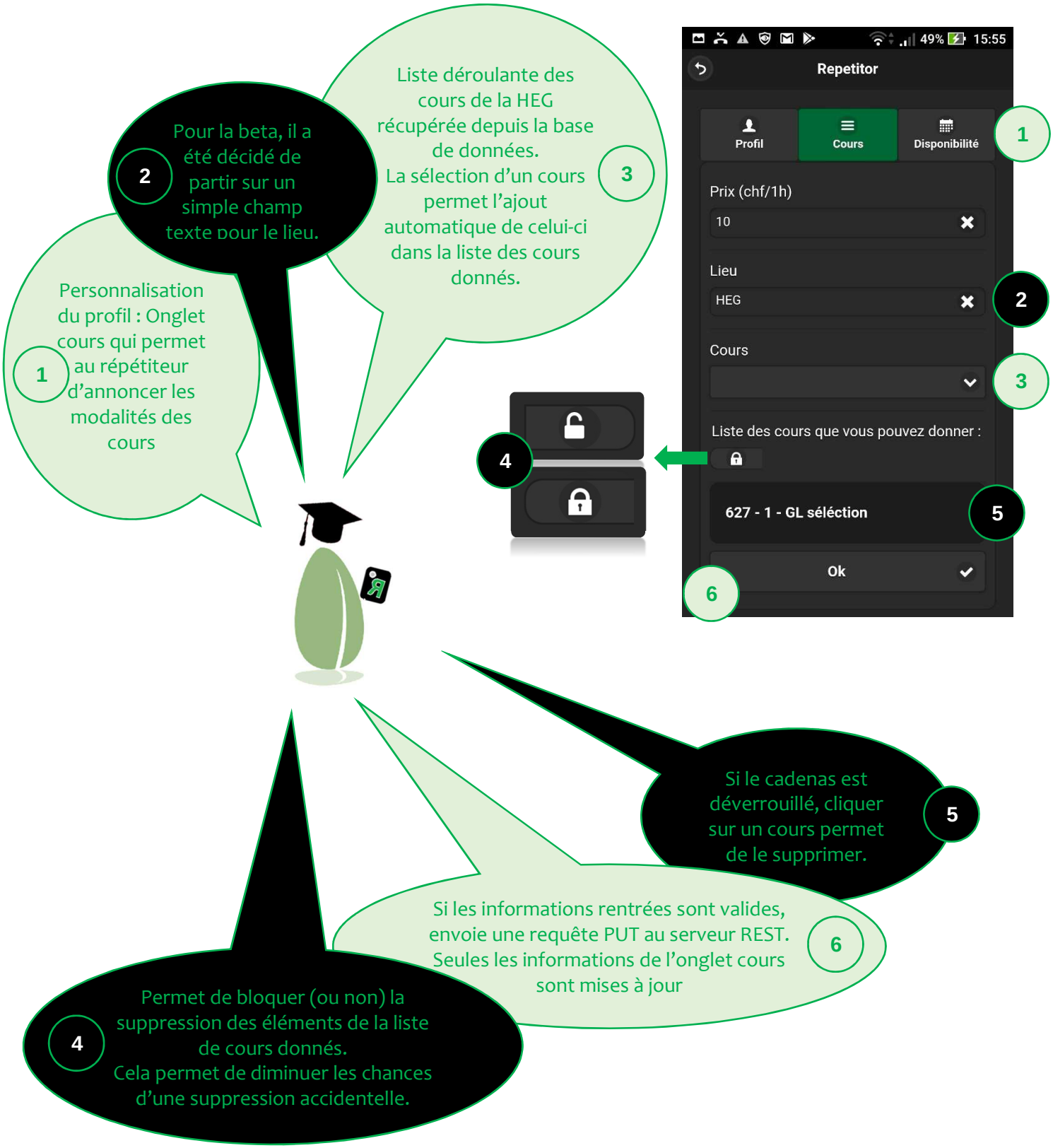
7.1.3 Interface visuel



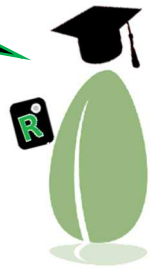





Personnalisation du profile : Onglet profil avec les informations générales liées à l'utilisateur.
1
L'adresse email ne peut être modifiée, le champ est donc bloqué.
2
Repetitor
Profil
Cours
Disponibilité
Prénom*
Diana
Nom*
Pinto
Adresse mail*
diana.pinto@outlook.com
Filière
Informatique de gestion
Site
3
Liste déroulante contenant les différentes filières de la HEG récupérées depuis le serveur.
De petites bulles d'informations contiennent des explications ou des conseils destinés à l'utilisateur.
4
Si les informations rentrées sont valides, envoie une requête PUT au serveur REST.
Seules les informations de l'onglet profil sont mises à jour
5
Vendez-vous avec une petite description !
Ok
Vous avez un portefolio en ligne, un blog ou un compte LinkedIn? Pourquoi ne pas l'utiliser pour vous démarquer.


Repetitor
Profil
Cours
Disponibilité
Prix (chf/1h)
10
Lieu
HEG
Cours
Liste des cours que vous pouvez donner :
627 - 1 - GL séléction
Ok
1
2
3
4
5
6
Personnalisation du profil : Onglet cours qui permet au répétiteur d'annoncer les modalités des cours
Pour la beta, il a été décidé de partir sur un simple champ texte pour le lieu.
Liste déroulante des cours de la HEG récupérée depuis la base de données. La sélection d'un cours permet l'ajout automatique de celui-ci dans la liste des cours donnés.
Permet de bloquer (ou non) la suppression des éléments de la liste de cours donnés. Cela permet de diminuer les chances d'une suppression accidentelle.
Si le cadenas est déverrouillé, cliquer sur un cours permet de le supprimer.
Si les informations rentrées sont valides, envoie une requête PUT au serveur REST. Seules les informations de l'onglet cours sont mises à jour

Afin de pouvoir communiquer des informations à l'utilisateur, j'ai réalisé quatre tamplates de messages :



Bravo !

Ceci est une information.

Attention, cette application est encore en cours de développement !

Une erreur a été signalé !

7.1.4 Scénarios

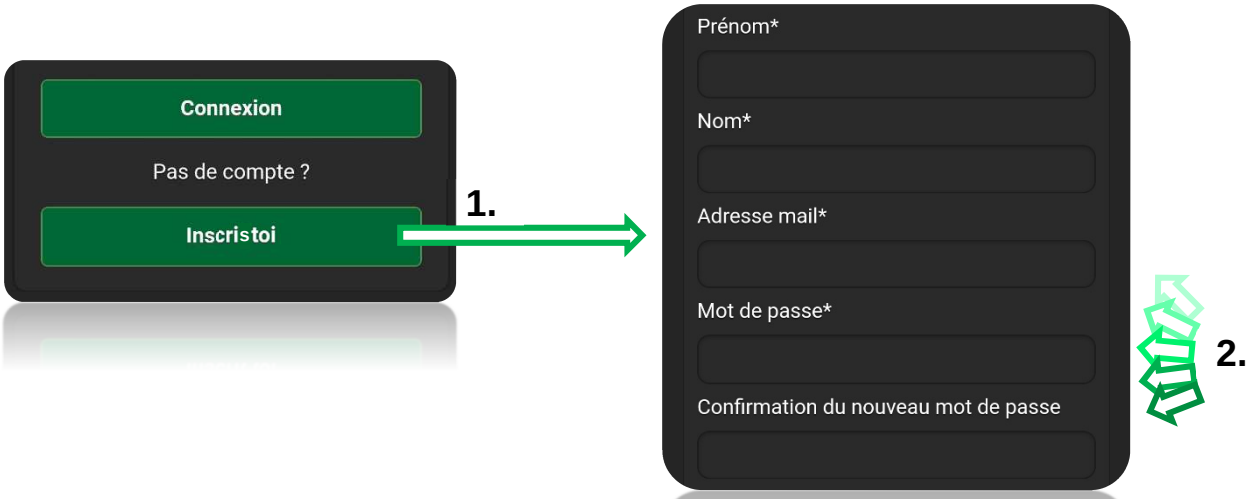
7.1.4.1 Scénario - Inscription

Situation : Un étudiant souhaite s'inscrire sur l'application.

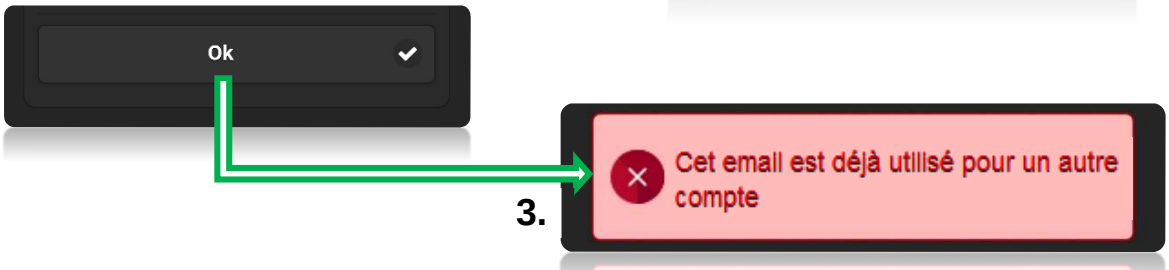
Prérequis : Avoir téléchargé l'application.

Détails :

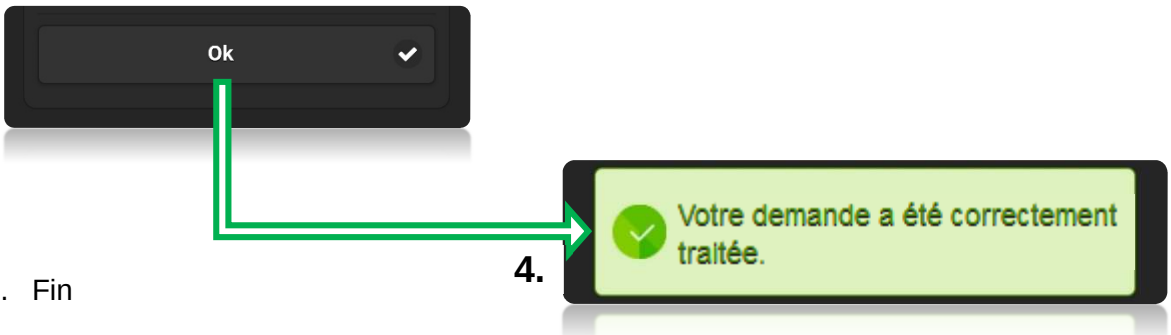
1. L'utilisateur va sur la page d'accueil et clique sur "Inscrit toi".
2. L'utilisateur remplit les différents champs obligatoires.



3. En cas d'erreur ou problème éventuel, l'utilisateur est informé avec un message adapté à la situation.



4. En cas d'erreur ou problème éventuel, l'utilisateur est informé avec un message adapter à la situation.



5. Fin

7.1.4.2 Scénario 2 : Configuration du profil

Situation : Un étudiant souhaite donner des cours, pour cela, il doit développer son profil afin de pouvoir être choisi comme répétiteur.

Prérequis : Avoir un compte.

Détails :

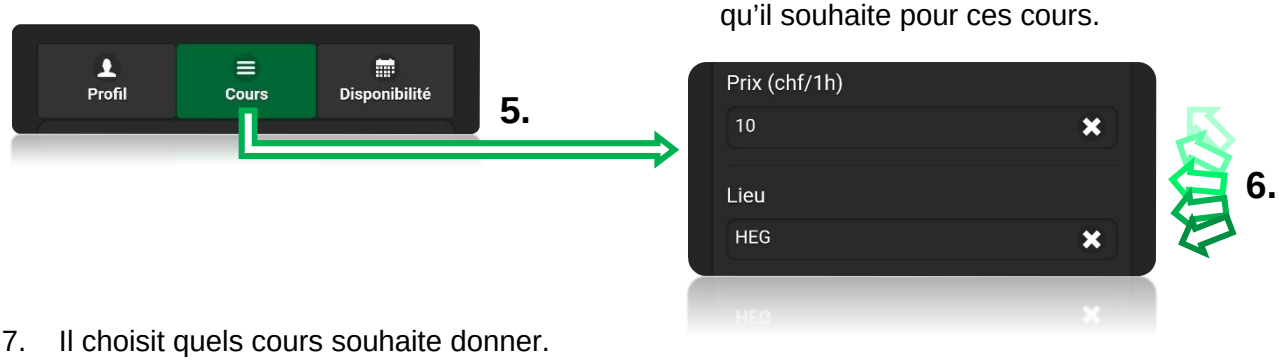
1. L'utilisateur ouvre la page de son profil.
2. L'utilisateur rentre les informations qu'il souhaite pour le profil.

3. En cas d'erreur ou de problème éventuel, l'utilisateur est informé avec un message adapté à la situation.

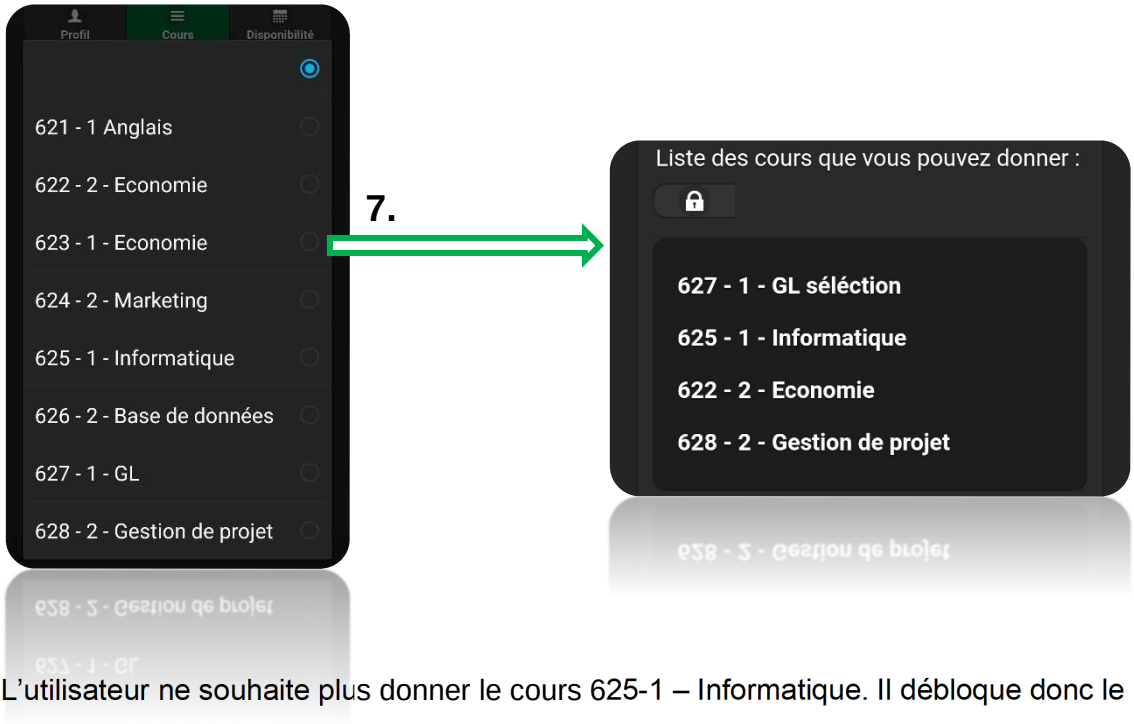
4. Dès que les informations du profil sont correctes, le profil est mis à jour.

5. L’utilisateur ouvre l’onglet des cours.

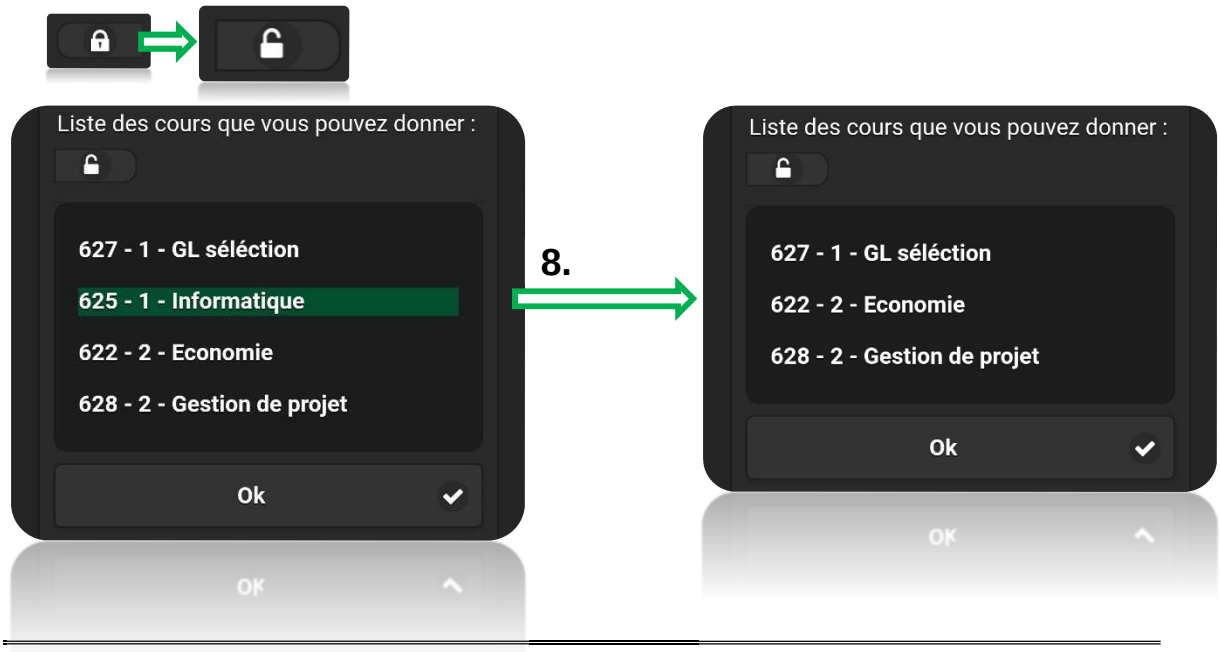
6. Il configure les spécifications qu’il souhaite pour ces cours.



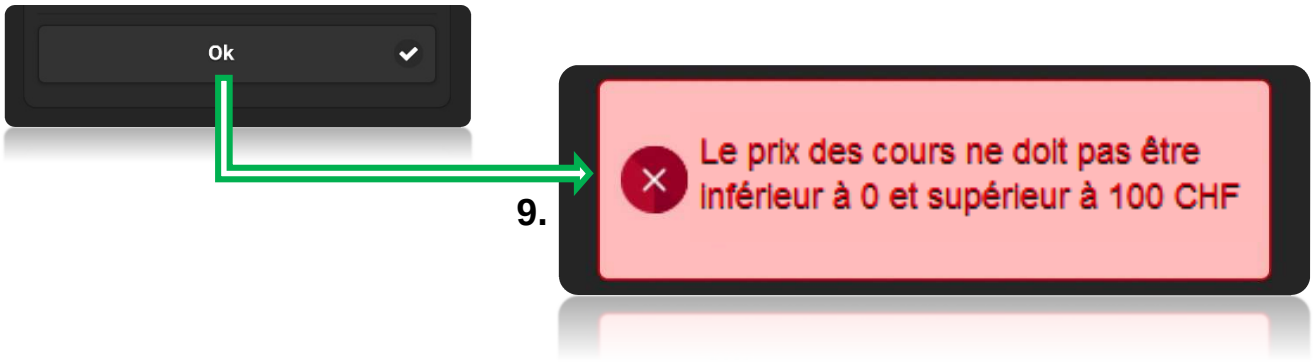
7. Il choisit quels cours souhaite donner.



8. L’utilisateur ne souhaite plus donner le cours 625-1 – Informatique. Il débloque donc le cadenas pour pouvoir le supprimer

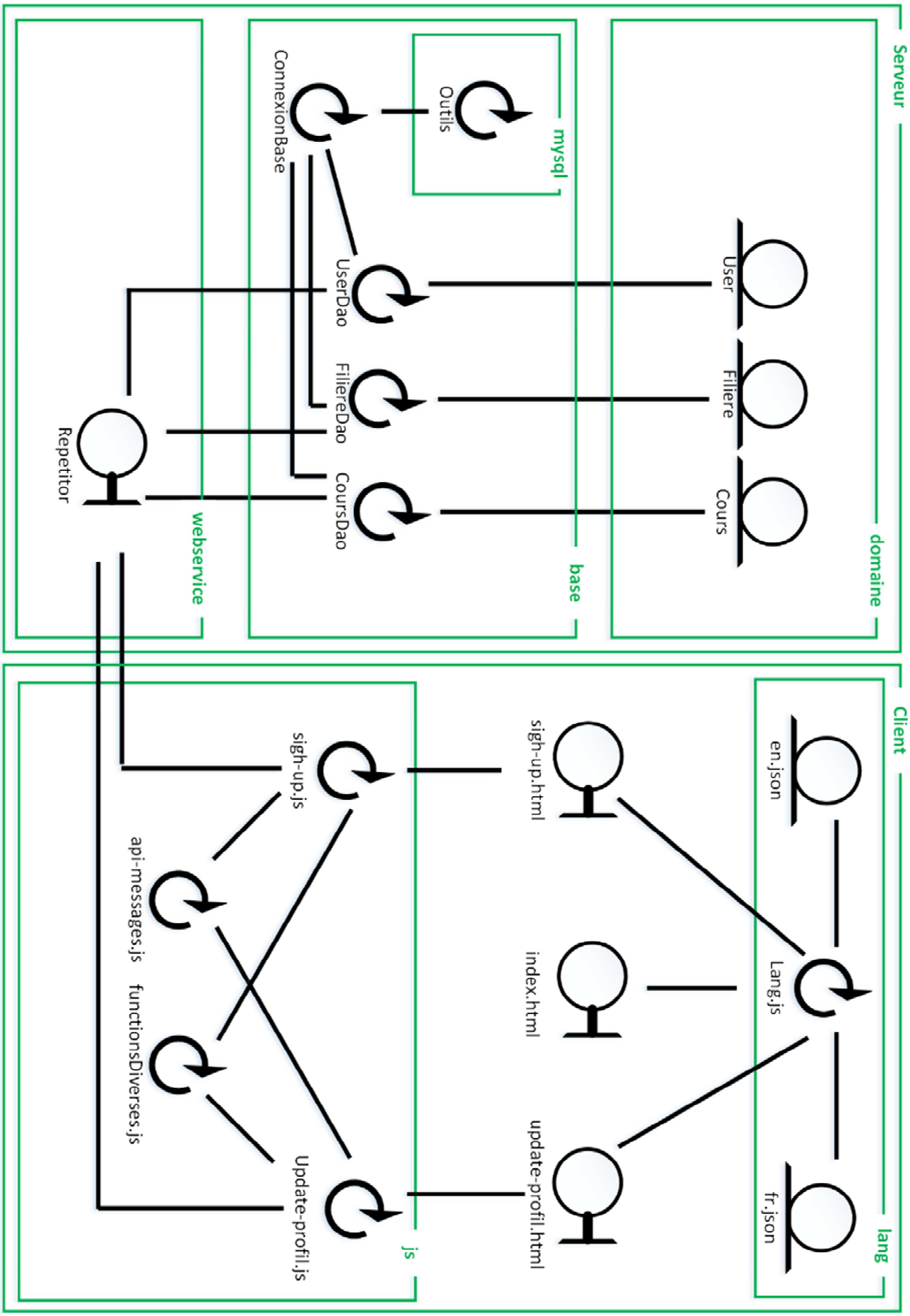


9. A nouveau, en cas d'erreur ou de problème éventuel, l'utilisateur est informé avec un message adapté à la situation.



10. Dès que les informations concernant les cours sont correctes, le profil est mis à jour.

7.1.5 Architecture du code



7.2 Fonctionnalités à développer

7.2.1 Disponibilité

Les répétiteurs peuvent inscrire leurs disponibilités dans le planning afin que les étudiants proposent des cours en fonction de celui-ci.

7.2.2 Recherche

Recherche simple permettant à un étudiant de consulter la liste des répétiteurs donnant le cours recherché. Cela permettra également de trouver un utilisateur en fonction de son nom / prénom.

7.2.3 Prendre / proposer un rendez-vous

À partir des disponibilités, un étudiant pourra proposer un rendez-vous au répétiteur choisi. Celui-ci, devra répondre dans les 24 – 36 heures afin de permettre à l'étudiant d'en rechercher un autre rapidement.

7.2.4 Consulter l'agenda

Depuis un agenda intégré, les étudiants pourront consulter les différents rendez-vous pris via l'application (En tant qu'élève et répétiteur).

7.2.5 Favoris

Une liste de profils favoris sera mise en place afin de permettre aux étudiants de retrouver leurs répétiteurs préférés plus rapidement.

7.2.6 Tchat

Un système de communication sous forme de tchat via l'application pourrait être mis en place pour faciliter la communication entre les étudiants.

7.2.7 Réputation

Pour permettre aux étudiants de choisir au mieux leurs répétiteurs, un système de notation et commentaires d'un profil pourrait être appliqué.

Avoir une bonne réputation → bonnes notes, permettra d'apparaître en priorité dans les listes de recherche.

7.2.8 Payement

Dans un premier temps, les paiements se feront entre les étudiants, l'application n'interviendra pas et Repetitor ne sera pas responsable en cas de non payement. Mais

par la suite, la possibilité de payer directement par l'application via, par exemple Paypal, sera sûrement mise en place.

7.2.9 Contact et proposition

Cette application est faite par des étudiants pour des étudiants, il est donc important d'écouter l'avis et les idées des utilisateurs afin d'améliorer l'application pour eux, pour nous. Une section de contacts et propositions permettra d'effectuer cela.

8. Conclusion

Tout au long de la durée de ce projet, en plus des diverses recherches effectuées, j'ai développé les fonctionnalités *Inscription* et *Configuration du profil*. :

Du côté client, à l'aide du Framework JQuery mobile, j'ai réalisé les maquettes html orientées mobile, puis j'ai intégré PhoneGap, créé la base de données MySQL, développé les Dao afin de récupérer depuis ma base de données, mis en place mes services web, réaliser les requêtes Ajax qui permettent de faire la communication entre le client et le serveur. Et bien sûr, intégrer la couche de sécurité sur mon serveur.

Ce fut une expérience très intéressante pour moi :

Tout d'abord, parce que nous avons un client web d'un côté et un serveur REST Java de l'autre. Ce sont deux technologies différentes et même si j'avais déjà travaillé avec chacune d'entre elles, ce ne fut pas facile de gérer les communications avec les requêtes Ajax. J'ai eu pas mal de problèmes surtout avec les contents types, par exemple.

Ensuite, il y a eu la couche de sécurité avec SSL/TLS. Cela fut un véritable défi à mettre en place étant donné que c'était la première fois. J'ai dû affronter de nombreux problèmes comme le fait que l'application ne puisse plus contacter le serveur à cause d'un problème de Cross-Origing Resource Sharing (CORS), puis plus récemment, PhoneGap qui refuse les requêtes vers un serveur avec un certificat auto-signé ce qui a pour conséquence de devoir travailler sur un serveur http durant le développement.

Tout cela fut très enrichissant et m'a beaucoup apporté. Je me suis même beaucoup amusée la plupart du temps car ce sont toutes des technologies qui m'intéressent, des défis que j'ai pris plaisir à relever et qui concernent une problématique traitée pour des étudiants par une étudiante.

Bibliography

Webpages

SIBERIAN, 2017. Siberian [en ligne]. [Consulté le 01.08.2017]. Disponible à l'adresse : https://www.siberiancms.com/

MOBINCUBE, 2017. mobincube [en ligne]. [Consulté le 01.08.2017]. Disponible à l'adresse : https://www.mobincube.com/fr/

DRUPAL, 2017. Drupal [en ligne]. [Consulté le 01.08.2017]. Disponible à l'adresse : https://www.drupal.org/

WORDPRESS, 2017. WordPress [en ligne]. [Consulté le 01.08.2017]. Disponible à l'adresse : https://fr.wordpress.com/

MORONY, Josh, 2017. joshmorony [en ligne]. 4 avril 2017. [Consulté le 28.07.2017]. Disponible à l'adresse : https://www.joshmorony.com/the-step-by-step-guide-to-publishing-a-html5-mobile-application-on-app-stores/

DOUDOUX, Jean Michel, 2016. jmdoudoux [en ligne]. 19.03.2016. [Consulté le 14.09.2017]. Disponible à l'adresse : https://www.jmdoudoux.fr/java/dej/chap-logging.htm

LOGBACK THE GENERIC, RELIABLE FAST & FLEXILBE LOGGING FRAMEWORK, 2017. LOGBack The Generic, Reliable Fast & Flexible Logging Framework [en ligne]. [Consulté le 15.09.2017].Disponible à l'adresse : https://logback.qos.ch/

APACHE, 2017. Apache [en linge]. [Consulté le 15.09.2017]. Disponible à l'adresse : https://logging.apache.org/log4j/2.x/

GRIESSER, Daniel, 2016. jCryption 3.1.0 [en ligne]. 30.01.2016. [Consulté le 21.09.2017]. Disponible à l'adresse : http://www.jcryption.org/#whattodo

CODRINGTON, Simon, 2016. sitepoint [en ligne]. 10.11.2016. [Consulté le 18.07.2017]. Disponible à l'adresse : https://www.sitepoint.com/12-amazing-jquery-tables/

GOOGLE CODE. Google Code [en ligne]. [Consulté le 02.08.2017]. Disponible à l'adresse : https://code.google.com/archive/p/crypto-js/

PHONEGAP, 2017. PhoneGap [en ligne]. [Consulté le 01.08.2017]. Disponible à l'adresse : https://phonegap.com/

L, Benjamin, 2014. alsacréations [en ligne]. 16.10.2014. [Consulté le 15.08.2017]. Disponible à l'adresse : https://www.alsacreations.com/tuto/lire/1646-introduction-phonegap.html

JQUERYMOBILE, 2017. *JQuery mobile* [en ligne]. [Consulter le 04.10.2017]. Disponible à l'adresse : https://jquerymobile.com/

JQUERY, 2017. *JQuery* [en ligne]. [Consulter le 04.10.2017]. Disponible à l'adresse : http://jquery.com/

JQUERYMOBILE, 2017. *JQuery mobile* [en ligne]. [Consulter le 01.10.2017]. Disponible à l'adresse : https://themeroller.jquerymobile.com/

JQUERYMOBILE, 2017. *JQuery mobile* [en ligne]. [Consulter le 04.10.2017]. Disponible à l'adresse : http://demos.jquerymobile.com/1.4.1/

DEFINITIONS MARKETING, 2017. *Définitions marketing* [en ligne]. [Consulter le 03.10.2017]. Disponible à l'adresse : https://www.definitions-marketing.com

WIKIPEDIA, 2017. Wikidedia [en ligne]. [Consulté le 13.10.2017]. Disponible à l'adresse : https://fr.wikipedia.org/

OPEN CLASSROOMS, 2017. *Open Classrooms* [en ligne]. [Consulter le 06.10.2017]. Disponible à l'adresse : https://openclassrooms.com/

W3SCHOOLS, 2017. W3Schools [en ligne]. [Consulter le 09.10.2017]. Disponible à l'adresse : https://www.w3schools.com/

STACK OVERFLOW, 2017. Stack Overflow [en ligne]. [Consulté le 11.10.2017]. Disponible à l'adresse : https://stackoverflow.com/

COEFFE, Thomas, 2017. blog du modérateur [en ligne]. 01.06.2017. [Consulté le 07.10.2017]. Disponible à l'adresse : https://www.blogdumoderateur.com/internet-trends-report-2017-kpcb/

MEEKER, May, 2017. KPCB [en ligne]. 31.05.2017. [Consulté le 07.10.2017]. Disponible à l'adresse : http://www.kpcb.com/internet-trends

DOUDOUX, Jean Michel, 2016. jmdoudoux [en ligne]. 19.03.2016. [Consulté le 14.09.2017]. Disponible à l'adresse : https://www.jmdoudoux.fr/java/dej/chap-json.htm

MRKALJEVIC, Mirnes, 2016. *Code Project* [en ligne]. 7 décembre 2016. [Consulté le 26.09.2017]. Disponible à l'adresse :

https://www.codeproject.com/Articles/1159537/Simple-JSON-REST-consumption-with-GSON-API

GITHUB, 2017. *GitHub* [en ligne]. [Consulté le 26.09.2017]. Disponible à l'adresse : https://github.com/google/gson

NODEJS, 2017. *NodeJS* [en ligne]. [Consulté le 20.08.2017]. Disponible à l'adresse : https://nodejs.org/en/about/

NODEJS, 2017. *NodeJS* [en ligne]. [Consulté le 20.08.2017]. Disponible à l'adresse :

https://nodejs.org/en/download/package-manager/

PANKAJ, 2017. *JournalDev* [en ligne]. 2 mars 2017. [Consulté le 25.09.2017]. Disponible à l'adresse : https://www.journaldev.com/498/jersey-java-tutorial

The Apache Sofware Foundation, 2017. *Apache Tomcat* [en ligne]. 29 septembre 2017. [Consulté le 10.10.2017]. Disponible à l'adresse : https://tomcat.apache.org/tomcat-8.0-doc/ssl-howto.html#Configuration

Yi Ming Huang, Dong Fei Wu, 2009. *IBM* [en ligne]. 24 septembre 2009. [Consulté le 27.09.2009]. Disponible à l'adresse : https://www.ibm.com/developerworks/library/wa-aj-tomcat/

APACHE CORDOVA, 2017. *Apache Cordova* [en ligne]. [Consulté le 11.10.2017]. Disponible à l'adresse : https://cordova.apache.org/docs/en/latest/guide/appdev/security/index.html

MATEI, Adrian, 2014. *CodingpediaOrg* [en ligne]. 05 avril 2014. [Consulté le 30.09.2017]. Disponible à l'adresse : http://www.codingpedia.org/ama/how-to-add-cors-support-on-the-server-side-in-java-with-jersey/

RAMON, Jorge, 2017. *DZone* [en ligne]. 17 mars 2015. [Consulté le 16.08.2017]. Disponible à l'adresse : https://dzone.com/articles/how-build-user-registration

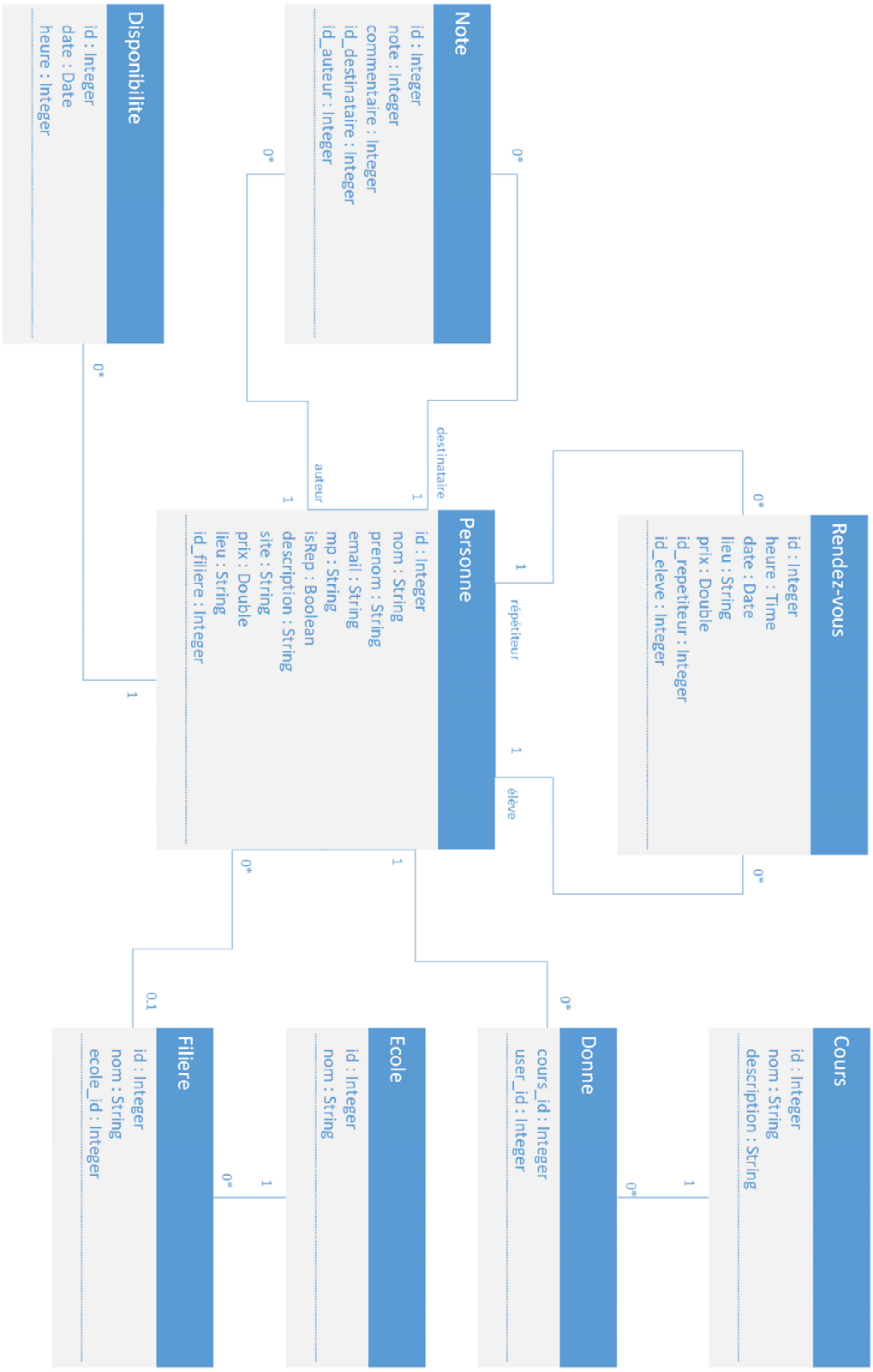
RAMON, Jorge, 2017. *DZone* [en ligne]. 05 août 2014. [Consulté le 16.08.2017]. Disponible à l'adresse : https://dzone.com/articles/mobile-app-tutorial-meeting

RAMON, Jorge, 2017. *DZone* [en ligne]. 10 avril 2015. [Consulté le 16.08.2017]. Disponible à l'adresse : https://dzone.com/articles/packaging-jquery-mobile

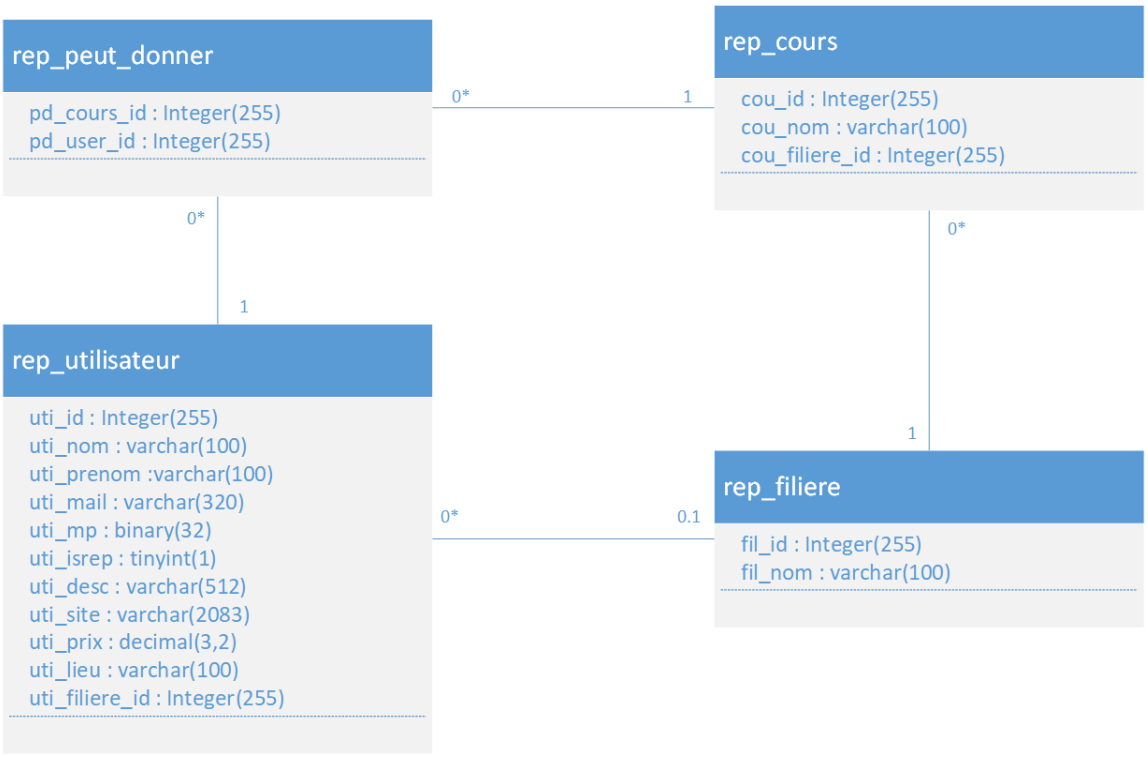
RAMON, Jorge, 2017. *DZone* [en ligne]. 05 juin 2015. [Consulté le 25.08.2017]. Disponible à l'adresse : https://dzone.com/articles/user-authentication-jquery

Annexes

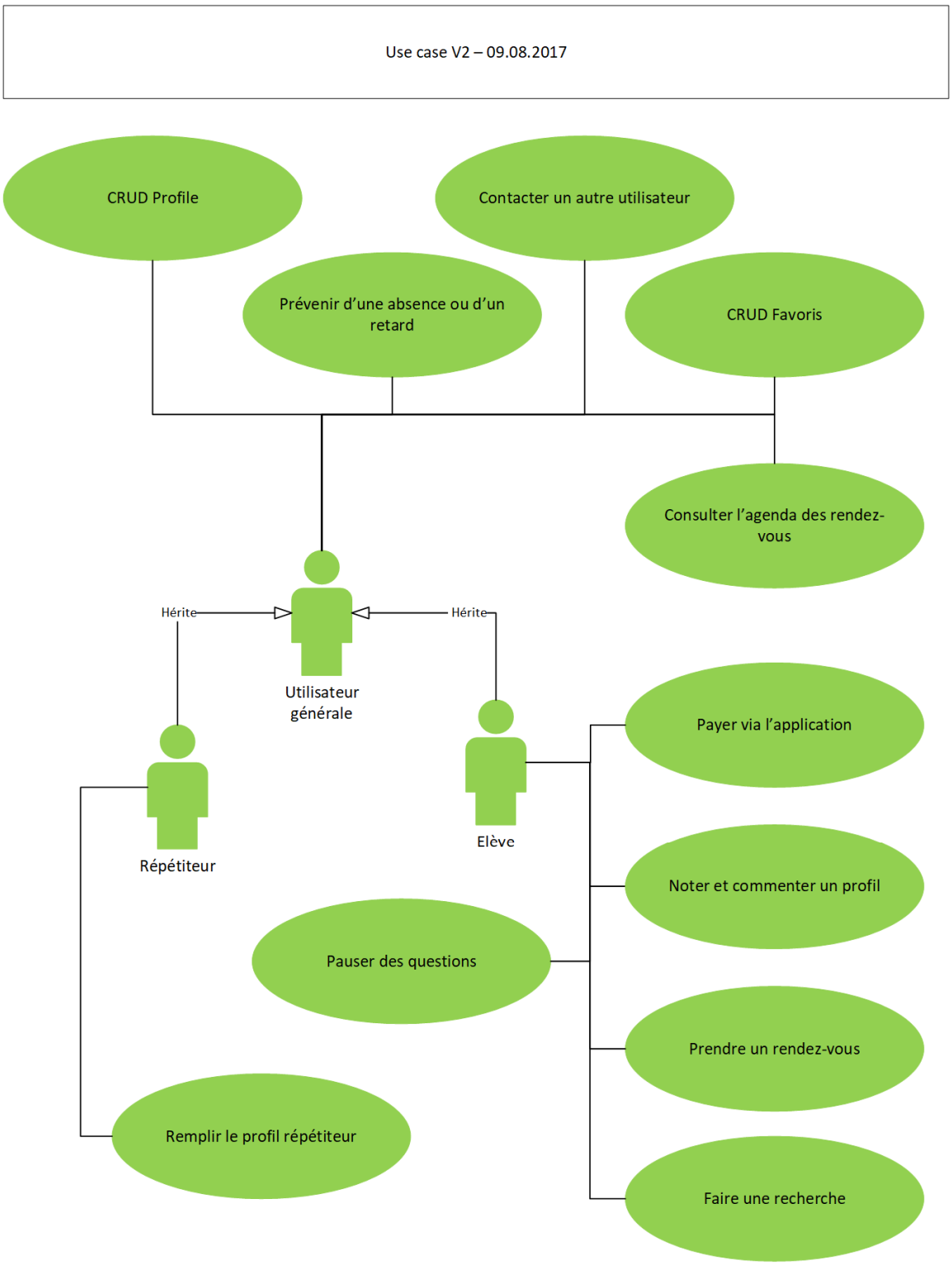
Annexe 1 – Ebauche du diagramme de classe



Annexe 2 – Diagramme de classe actuel



Annexe 3 – Use Case



Annexe 4 – Les Works flow identifiés